IN THE DRAWINGS:

Formal drawings have previously been supplied.

REMARKS

In regard to the Examiner's Office Action of January 26, 2006, Applicants have now reviewed the specification and claims and will present certain clarifying amendments thereto.

Regarding the print-out of the application which indicates the letter "e" is missing, or barely visible, Applicants will herein be presenting another copy of the specification and claims which will take care of the missing letters and poorly-printed areas. Attached hereto is another copy of the original specification. Likewise, another copy of the original claims are being presented with the specification in order to correct the informality of the missing letter "e" in the first line of each of the mentioned claims.

In regard to Examiner's rejections for non-statutory subject matter under 35 USC Article 101, Applicants have now amended the claims to indicate structural limitations that would define the system as a tangible device or method of using such device. Thus, the methods are now implemented as indicating working on a computer or having a software product stored in a computer-readable medium that operates the specified method when being executed on a computer.

Examiner has rejected claim 12 for indefiniteness under 35 USC, Article 112, second paragraph, where the phrase "such as" renders the claim indefinite. This has now been corrected in order to make claim 12 in a more definite and expressible format.

Examiner has rejected claims 1-3 and 13-15 for anticipation under 35 USC 102(b) --- as being anticipated by the Collins Reference "U" entitled "*Parallel And Sequential Job Scheduling In Heterogeneous Clusters; A Simulation Study Using Software In The Loop*". This has been published by the High-Performance Computing And Simulation Research Laboratory, College of Engineering, University of Florida.

The Collins reference involves a parallel job scheduling algorithm for heterogeneous computing and the application of SWIL (software-in-the-Loop) simulation to the analysis of job scheduling systems. This involves the design and optimization of job scheduling algorithms for heterogeneous computing environments which requires the ability to predict scheduling performance. In Collins, there are experimentally determined models for the prediction of job execution times on both sequential and parallel computing resources -- these are combined with the implementation of a new scheduling algorithm and a software-in-the-loop (SWIL) simulation.

Applicants would herein traverse Examiner's considerations that the Collins Reference "U" is applicable to Applicants' system and configuration. *Please see the comments in Appendix I.*

> **Examiner has further rejected claims 4-11 and 16, for anticipation under 35 USC, Article 102(b), as anticipated by the Snelick Reference "V", which reference is entitled *"S-Check: A Tool For Tuning Parallel Programs"*. Here, Applicants would traverse the Examiner's considerations that the Snelick reference actually applies to the configuration of Applicants. *Also see the comments re Snelick in Appendix I.***

Snelick provides a tool called "S-Check" for identifying performance bottlenecks in parallel and network programs. It predicts how refinements in parts of a program are going to affect performance by making local changes in code efficiencies and correlating these against overall program performance.

Examiner has rejected claims 17 and 18 for obviousness under 35 USC, Article 103(a), as being unpatentable over the Snelick Reference V *("S-Check, A Tool For Tuning Parallel Programs")*.

Here, it is noted that Snelick does not expressly disclose the system of claim 16, where said means (e) to condition delays includes:

> (e1) means to calculate said delay as a delay parameter using a fuzzy logic method to optimize said processing.

Here, Examiner says he takes official notice that fuzzy logic is well-known in the art of computer programming. However, here it must be indicated that this particular element is only one phase of an overall system, and as such, the fact that it is well-known in the art does not invalidate an entire claim which involves several other means facilities.

At first, it might be of value to note the major differences between Applicants' systematic approach and that of the Collins Reference "U":

(This also applies to the Snelick Reference "V").

The major differences that are observable between Applicants' configuration and the Collins-Snelick references are as follows:

> (a) Collins and Snelick are batch/job oriented. Quite contrarily, Applicants' approach is based on a transactional/method approach.
>
> (b) Applicants' system provides for fine-grained control over the business methods of an application, (that is to say, buy, sell, browse), while contrarily, the Collins and the Snelick references are involved with job-level service-level optimization.
>
> As a special differentiation, Applicants' "historical database" is based on component method interactions. The Collins-Snelick situation is based on "previous" job execution histories.

(c) The Collins software and loop operations (SWIL) is an off-line simulation technique. Quite contrarily, Applicants' approach is to inject code into the RUNNING application.

(d) The delays which are inserted in the Snelick Reference "V" are "perturbation delays" that are used to shake-up the system to see how it operates in different environments. Snelick uses these for experimental purposes.

> Note, however, the operating features where Applicants' delay operations are intended to improve the performance of other methods that are deemed to have higher priority, and in some cases, improve the overall performance of the running application or another application when a system is thrashing.
>
> The delays in Applicants' system are intended to change the execution profile of the applications.

Applicants' system and method involves the maintenance of target response times in business methods, and as was noted in the drawing of Fig. 6, where there is an indicative factor which provides information as to the response time involved and how a response time can be improved.

It is emphasized that Applicants will inject code into the "running" application, which does not require an off-line simulation technique, as does Collins and Snelick.

Note that Snelick shakes-up the system to see how it operates in different environments as done for experimental purposes. Quite contrarily, the delays provided by Applicants' system are intended to improve the performance of methods deemed to have a higher priority and this way improve the overall

performance of the application, or another application when the system is thrashing about trying to decide which application should get greater priority. Thus, Applicants' system changes the execution profile of the applications.

In regard to claims 17 and 18 which have been rejected for obviousness on the Snelick Reference "V", plus Applicants' recognition of fuzzy logic as being well-known, it should be noted in the following way:

> The delays involved in the Snelick reference are "perturbation delays" that are used to de-stabilize the system. Quite contrarily, Applicants' delays are intended to affect the execution profile of the system, that is to say, giving users who are buying items a priority over users, for example, who are just browsing. Applicants also increase the scaleability of their application when the server is overloaded by the manner of introducing certain delays.
>
> Again, as mentioned above, the use of fuzzy logic is not a critical factor of this system, but it is used to calculate certain values which are cited in the claims. It is just one factor in a number of factors which define the overall system.

There are a number of problems that are handled by Applicants' configuration which are not addressed by the references "U" and "V".

> Applicants' deal with interactive Web-based, transactional applications, and then exert a fine-grained control over the method level execution of these applications. The Collins reference deals with batch jobs, or entire applications at the response time level. The Snelick reference deals with parallel executions of computer code.

Note that Applicants exert control over the method level execution of the applications which are interactive, Web-based, transactional applications.

Applicants' system and method provides the ability to determine interactions between methods of a component-based system and then to use that information to affect the behavior of the application at a meaningful business level. This is not the characteristic teaching of either Collins or Snelick.

In view of the above-mentioned factors, Applicants have amended the claims of their configuration in order to better differentiate those working factors which are not taught or could not possibly be inferred from the Collins and the Snelick references.

It should be indicated that the claims of an application should be evaluated as a whole in their entirety, and not just on the basis of one or two individual clauses which might have some relationship to other cited references. Thus, while looking at Applicants' claims as a whole in their entirety, it should be noted that there is a combinative configuration which involves factors which cannot be taught by the cited references, and in this regard it is respectfully requested that Examiner consider the claims as a whole in their entirety and provide a timely Notice of Allowance therefor.

Respectfully submitted,

By Alfred W. Kozak

Alfred W. Kozak
Reg. No. 24,265
(858) 451-4615
(949) 380-5822

Certificate of Mailing (37 CFR 1.8a)

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being deposited with the United States Postal Service on the date shown below with sufficient postage as first class mail in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Patti S. Preddy

Date: April 14, 2006 Patti S. Preddy

APPENDIX I
DOCKET AWK02-004
USSN 10/670,976

Some further notes may be in order in regard to the Collins reference "U", which is entitled "*Parallel And Sequential Job Scheduling In Heterogeneous Clusters: A Simulation Study Using Software In The Loop*".

Here, it should be noted regarding the task of designing for and optimizing job scheduling algorithms for heterogeneous computing environments -- where the article indicates that this requires the ability to predict scheduling performance.

Thus, there are a number of experiments required in Collins to sketch scheduling paradigms. This requires experimentally determined models for the prediction of job execution times on both (i) sequential and (ii) parallel computing resources which can be combined with a scheduling algorithm and a software-in-the-loop (SWIL) simulation.

As a result, the author Collins and George developed a number of case studies in order to determine policies for overhead, scheduling performance and the impaction of pre-emption and priority policies. This involved: (i) a Parallel Job Scheduler (PJS) algorithm; and also (ii) whether and when to use a Sequential Job Scheduler (SJS).

One goal of their research was to develop a unified approach to parallel and sequential job scheduling for heterogeneous clusters, and then the possible development of a Parallel Job Scheduler, a scheduling algorithm.

Collins then sets-up a number of experimental situations using five job classes for test cases to reflect a reasonably diverse set of computing requirements.

The Collins priority system was using three priority levels - high, medium and low. For example, a first priority distribution had 10/10/80 which meant 10% high priority jobs, 10% medium priority jobs, and 80% low priority jobs. Another distribution was made for 25/25/50. Another distribution was made as 50/25/25.

After all these various experimental situations were run in order to find more efficient levels of job scheduling, a number of very complex conclusions were derived.

Finally, a number of conclusions were made from each of the case studies.

One conclusion was that the Parallel Job Scheduling algorithm had significantly more ability to favor high priority jobs over the Sequential Job Scheduling algorithm.

Another conclusion is that the PJS algorithm used in pre-emptive priority mode of operation is effective at favoring the high-priority jobs over the lower-priority jobs for a pre-emptive priority-based scheduling system. However, there does appear to be a performance penalty in the aggregate sense associated with the use of preemption and priority under these conditions, but the penalty is not generally very large.

Then, the article by Collins and George further indicates that much more research is possible, and various areas of directions to investigate can be made in the future to see whether there could be any more improvement developed in performance and on scheduling overhead. And then another possible direction and research in the future is to further investigate

the possibilities and trade-offs in a pre-emptive scheduling scheme.

While all of this research and speculation is very interesting, it still does not provide the capabilities which are provided in Applicants' system, whereby component-based transactional applications are optimized in a component balancer system. The Collins reference does not set-up specified response times as a goal for a specific group of methods. Collins does not provide any possibility for setting-up of a no-delay in a method to a maximum delay in a method. Further, there is no concept of calculating statistical metrics between "pairs" of the transactional methods involved (A,B,C. . .N).

Further, the Collins-George reference "U" does not provide for the automatic optimization of the processing sequence of transactional applications, nor does it allow for manually optimizing the processing sequence.

Further, there is no adjustment means taught by the Collins reference "U", where a delay increment is provided according to the load on the system as sensed by the number of calls per-second.

Regarding the Snelick reference "V" entitled "*S-Check: A Tool For Tuning Parallel Programs*", the S-Check is based on techniques of Synthetic-Perturbation Screening (SPS). This is a diagnostic technique employing artificial code (delays) placed within segments of a parallel program. Then, here there is provided screening techniques based on statistical experimental design to pinpoint those program segments which are most sensitive to perturbation (delay). Thus, a subset of the program segments so identified --- can then be candidates for improvement.

The result here is a portable and scaleable technique for detecting performance bottlenecks in parallel programs. In the Snelick case, a design of experiments is done where a plan uses a complete description of a minimum step of perturbation patterns needed to carry-out a meaningful program investigation.

It should be noted in Applicants' system, there is no such requirement for perturbation patterns or making experimental researches in this matter. Snelick seeks to do an examination of code segments to find candidates for bottlenecks and these are identified as potential problems. Then, Snelick makes a chart shown in his Table 1 which is a sensitivity analysis of an image benchmark. This is to find the dominant bottleneck in the architecture.

After analyzing the various types of bottlenecks, then it is possible to try to provide solutions for these bottlenecks.

None of these operations in Snelick could possibly teach or provide for the features shown in Applicants' claims, and which were also enumerated in the discussions regarding the Collins reference.